報告番号	※甲　　第　　号

主　論　文　の　要　旨

論文題目　組込みシステム開発のためのドキュメント・ソースコード分析支援

氏　名　山 本 椋 太

論　文　内　容　の　要　旨

近年，組込みシステムの大規模化・複雑化が進行しており，開発効率の向上が求められている．また，組込みシステムのライフサイクルは短いため，短納期化の要求があり，開発の効率化に対する要求がある．開発の効率化の手法として，手戻りの抑制や，ソフトウェアの再利用があげられる．これらの開発の効率化のためには，再利用が困難なソフトウェアや，ドキュメントの低品質化が問題となる．そこで，これらの問題によってについて述べる．

まず，ソフトウェアの再利用について，開発現場では，ソースコードのレガシー化が生じていることがある．ここでレガシー化したソースコード（レガシーコード）とは，開発者が不在となり，かつ開発関連文書の保守が不十分であるような状況となったとき，内容を理解することが困難となったソースコードである．レガシーコードを用いて開発する開発者は，ソースコードの内容の詳細を把握していないが，入出力情報やテストケースなどは残っていることを仮定しており，それらからレガシーコードをブラックボックスとして扱うことはできる．そのため，レガシーコードを用いる場合には，アルゴリズムを変更するのではなく，分岐を増やすことで対応することがある．その結果として，ソースコードの複雑度が向上することとなり，さらに保守性の低下につながる可能性がある．

次いで，ドキュメントの低品質化について述べる．開発者がそれらの文書を自然言語によって記述することで，曖昧表現や誤り表現を生じる可能性がある．自然言語によって記述された文書に存在する曖昧表現や誤り表現は，システム開発の手戻りの要因となることがあり，修正されることが望ましい．すなわち，要求仕様書はシステム開発の最序盤で作成される文書であり，この時点での曖昧さや誤りは後の工程に対して悪影響を及ぼし，手戻りの原因となりうる．また，組込みシステムの要求仕様書は，エンタープライズシステムの要求仕様書と比べて，詳細な情報が書

かれていることが多い．エンタープライズシステムの要求仕様書中の誤り・曖昧表現の抽出や調査の研究は存在するが，状態遷移記述や設計情報などが存在することが多い組込みシステムの要求仕様書における，誤り・曖昧表現の種別や出現数が明らかではない．以上より，本論文では組込みシステムの要求仕様書について着目する．実際に，要求仕様書から誤り・曖昧表現を抽出できればよいが，そのためには，抽出するべき誤り・曖昧表現が明らかになっている必要がある．

以上より，本研究では組込みシステム開発の効率化に向け，次の 3 つの研究を行う．(**研究 I**) ソースコードとの対応が取りやすいモデルである細粒度状態遷移表（Micro State Transition Table, MSTT）の提案，(**研究 II**) C 言語ソースコードから MSTT の抽出を支援するツールの提案，(**研究 III**) 要求仕様書中の誤り・曖昧表現の調査および (**研究 IV**) 要求仕様書中の誤り・曖昧表現の調査，以上の 4 つの研究を実施した．

研究 I では，ソースコードとの対応が取りやすいモデルである MSTT の提案を行う．本研究では，MSTT の定義を行い，ケーススタディを実施する．本研究の関連研究にあるように関数呼び出しをイベントや状態値として扱うのではなく，MSTT は，条件分岐文をイベントや状態値として扱う．MSTT は，組込みシステムの状態遷移表を表現することを目的としており，組込みシステム特有の非同期なイベントが出現することを考慮し，MSTT の上の行から下の行に向かってイベントを評価するように表現している．また，関連研究では状態分割ポイントや状態変数を指定することで，組込みシステムのソースコードから状態遷移モデルを作成可能な手法があるが，状態分割ポイントを指定するためには，ソースコード全体を読み，状態分割ポイントを判断する必要がある．イベントや状態値判定が，従来の状態遷移表と同様に一意なもののみとするためには，状態分割ポイントを指定するか，または元のソースコードを書き換える必要がある．これに対して，本研究における MSTT は，ソースコードを変更せず，かつ状態分割ポイントを判断する必要がない．従来の関数呼び出し単位の状態遷移表と比べて大規模になることが多いが，ソースコードとの対応はわかりやすく，詳細な処理の記述を可能としている．表の規模は大規模なものになるが，ソースコード上の 1 つのパスの条件をまとめてイベントを独立させた表と比べて 1 つのイベントの条件式の項数は低減しうる．ケーススタディでは名古屋大学の学生 4 名が，ソースコードから MSTT の抽出を試みた．このとき，ソースコードから MSTT を抽出するプロセスや MSTT そのものから，ソースコードへの理解を深めることができた．しかし，手作業による MSTT の抽出は難度が高くかつ時間がかかる．すべての参加者は，手作業において 1 度では正確に MSTT を抽出できず，抽出結果を修正する必要があった．これらの結果から，MSTT の自動抽出または抽出支援を行うツールが必要であると考えた．

そこで，**研究 II** では，MSTT の抽出支援ツール RExSTM の開発に取り組んだ．

RExSTM は，C 言語ソースコードを入力としてソースコード解析を行い，状態遷移表抽出を支援するツールである．ユーザは，RExSTM を用いた MSTT の抽出過程において，ソースコード中の状態を表す変数（状態変数）をツールに与える必要がある．状態変数候補の抽出は自動化されているが，MSTT として有効な表現となるような変数は，ユーザが選択することとしている．RExSTM を用いた MSTT の抽出と，手作業による作成時間を比較すると，ツールの有無において有意差があると認められ，RExSTM による抽出支援の有効性があると結論づけた．

研究 III と**研究 IV** は，組込みシステムに関する要求仕様書の分析である．まず**研究 III** として，要求仕様書の誤り・曖昧表現の実証的調査である．要求仕様書中に存在する誤り・曖昧表現を含む問題を定義し，組込みシステムについての実際の要求仕様書中に存在する問題の分析を試みた．組込みシステムの要求仕様書は，エンタープライズシステムの要求仕様書と比較して，クライアントもドメイン知識を有していることが多く，設計情報が含まれていることが多い．そのため，後述する**研究 IV** に関連して，記述対象システムの状態遷移に関する記述が存在することもある．調査の結果，ドメイン知識への依存が弱い問題については，ドメインの専門家でなくとも検出することができた．そして，最も検出数が多い「構文の曖昧表現と一貫性の問題」を 6 つに細分化し，細分化された問題の検出手法として，精度を向上する手法と，再現率を向上する手法の 2 つを検討した．これらの手法を実装して評価した結果，構文上の曖昧の検出数が最多であった．また，手法の適用結果から，字句・構文解析によって検出可能な問題は，高い精度・再現率で検出可能だとわかった．

研究 IV，先行研究において開発された自然言語で記述された要求仕様書から状態遷移表を抽出支援するツールの機能の内，節の分類機能の評価を行い，その結果から誤り・曖昧表現の検討を行った．このツールは，インタラクティブな操作に基づき，ツールの入力制約を満たした要求仕様書から，状態遷移表の抽出を支援する．入力した要求仕様書は，**研究 III** でも用いた組込みシステムの要求仕様書であり，上述の通り，エンタープライズシステムと比べて詳細な記載がある事が多い．そのため，状態遷移に関する記述も存在しており，先行研究におけるツールは，組込みシステムの要求仕様書が持つこの特徴に注目している．状態遷移表を抽出する過程において，このツールは独自定義の解析木を抽出し，抽出された解析木を用いて要求仕様書中の各文中の節に対して，**条件節**，**処理節**および**定義節**のいずれかの分類を与える．これらの分類精度を調査し，このツールの分類能力を確認するとともに，分類できない節が存在しないかを確認することで，誤り・曖昧表現の検出につながらないかを確認した．その結果，分類精度は，F 値で評価した場合に全体として 0.9 を超える結果となった．しかし，インタラクティブな操作において，処理節か定義節かを分類することが困難な節が存在することがわかった．解釈を機械的

に決定することができないために，インタラクティブな操作を必要とするため，インタラクティブな操作を必要とする箇所は曖昧表現である可能性があると考えた．

以上4つの研究により，組込みシステム開発を効率化するためのソースコードや要求仕様書の解析に基づく技術的負債を発見する支援，および識別を行った．